Towards a Cyber Defense Framework for SCADA Systems Based on Power Consumption Monitoring *

Jarilyn M. Hernández *
Lane Dept. of Computer Science and Electrical Engineering
West Virginia University
jhernan7@mix.wvu.edu

Qian Chen, Chelsea Calhoun, and Summer Sykes
Department of Engineering: Computer Science Technology
Savannah State University
chenq@savannahstate.edu

Jeffrey A. Nichols
Cyber and Information Security Research Group
Oak Ridge National Laboratory
nicholsja2@ornl.gov

Abstract

Supervisory control and data acquisition (SCADA) systems are industrial automation systems that remotely monitor and control critical infrastructures. SCADA systems are major targets for espionage and sabotage attackers. Current commercial off-the-shelf security solutions are insufficient in protecting SCADA systems against sophisticated cyber-attacks. Furthermore, these breaches are not detected in real-time or fast enough to prevent further damages. To address this challenge we present a feasibility study that proves monitoring power consumption of SCADA devices is an effective approach to detect cyber-attacks. We built a testbed containing a Programmable Logic Controller (PLC) that was instrumented to record its power usage. Three SCADA-specific cyber-attacks were simulated and we report the power consumption of the PLC under these normal and anomalous scenarios. We show that it is possible to distinguish the PLC power utilization between these scenarios. In route to this result we found and describe vulnerabilities in the DF-1 protocol.

1. Introduction

SCADA (Supervisory Control and Data Acquisition) systems refer to control systems that are used to monitor and control enterprise-wide operations of large-scale and distributed critical infrastructures. For example, SCADA systems are widely used to monitor electrical assets (i.e., substations and transformer), telecommunication facilities (i.e., switches and transmitters), water and waste equipment (i.e., water pumps and valves) and oil and gas pipelines (i.e., gas pumps and valves). Usually SCADA systems contain a set of devices such as sensors, actuators, programmable logic controllers (PLCs), remote terminal units (RTUs), human machine interfaces (HMIs), and master terminal units (MTUs).

Attackers can compromise SCADA systems by exploiting vulnerabilities residing in security policies, software, communication channels, wireless channels, and social engineering. Recently, SCADA systems have become a target for cyber-attacks, according to a recent report from the Organization of American States (OAS) [1] , due to their potential impacts on properties, economies, and human lives. The annual threat report from Dell shows that cyber-attacks against SCADA systems increased from 91,676 to 163,228 between January 2012 to January 2013 and increased to 675,186 at the end of January 2014 [2].

Specific, recent examples producing widespread damage include:

- In 2000, a disgruntled employee compromised the Maroochy Shire (Queensland computerized waste management system) causing millions of gallons of sewage spill into waterways, local parks and rivers [3].
- In 2010, a worm called Stuxnet [4] was detected in Iranian uranium enrichment facilities. This worm increased the operating speed of the Iranian IR-1 centrifuge from 1,064 Hz to 1,410 Hz for fifteen minutes before returning to its normal running frequency [5, 6]. This incident temporarily derailed Iran's nuclear program by destroying roughly 1,000 centrifuges.
- In 2013, a cyber-espionage group known as "Dragonfly" used phishing sites and Trojans to compromise more than 1,000 energy supplier organizations in Europe and North America [7].

In this paper, we demonstrate a new approach to monitor and analyze the power consumption of a SCADA device (e.g., PLC) in order to detect SCADA-specific cyber-attacks. Our primary goal is to prove that there are detectable differences in the power profiles between normal behavior and behavior when under cyber-attacks. We describe a SCADA testbed that enabled us to monitor and collect the power consumption of a PLC under normal operations, command injection attacks, denial of service attacks, and replay attacks.

The rest of this paper is organized as follows. Section 2 presents a literature review on detection tech-

* This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan). Research sponsored by the Laboratory Directed Research and Development Program of Oak Ridge National Laboratory, managed by UT-Battelle, LLC, for the U. S. Department of Energy.

* Also affiliated with the Cyber and Information Security Research Group at Oak Ridge National Laboratory

URI: http://hdl.handle.net/10125/41508
ISBN: 978-0-9981331-0-2

niques for identifying SCADA-specific cyber-attacks. Section 3 describes the experimental design (hardware and software configuration). Section 4 introduces the approach to collect PLC power consumption data. Section 5 validates our proposed approach by analyzing power consumption data under various scenarios. Finally, we present our conclusions and suggestions for future work in Section 6.

2. Related Work

The SCADA-specific cyber-attacks introduced in Section 1 show that traditional prevention mechanisms such as firewalls and anti-virus software are not enough to protect SCADA systems against sophisticated cyber-attacks. Researchers therefore are focusing on specific techniques to defend SCADA systems in depth. For example, Chen et al. [8, 9] adopted the idea of autonomic computing and created self-protecting systems that can estimate known attacks before they impact devices in the network. If an attack bypasses an Intrusion Detection System (IDS), a second line of defense provides patterns of known attacks to an Intrusion Response System (IRS). A controller in the IRS evaluates candidate protection mechanisms considering pre-defined criteria and then selects the optimal protection mechanisms to regulate system behavior back to normal. Unknown attack patterns were investigated by an online-learning module and were sent to the IRS similar to the known attack scenario. As the result, the self-protecting systems were validated to protect SCADA systems against both known and unknown attacks in near real-time with little or no human intervention.

Rule-based signature techniques are commonly used to detect SCADA-specific cyber-attacks. Particularly, this technique is often adopted to detect those cyber-attacks that interrupt, fabricate, intercept or modify communication messages between the HMI and PLC. H. Bao et al. [10] proposed a behavior rule-based methodology monitoring devices in the smart grid for insider threat detection. Results showed that this approach could detect abnormal behaviors in pervasive smart grid applications. However, the number of rules for detecting polymorphic and metamorphic malware can be extremely large. Thus, a more efficient approach must be developed to speedup the detection and enhance accuracy.

The objective of this research paper is to prove the concept that cyber-attacks can be detected by using power usage profiles. Similar power-based cyber-attack detection techniques have emerged for detecting cyber-attacks in other domains (or computing systems) by collecting power profiles. For example, smartphones with in-band power collection [11], medical devices with AC power [12], and software defined radio [13].

J.Hoffman et al. [11] tried to detect smartphone malware by analyzing the power consumption of the device. However, this approach failed due to the noise caused by unpredictable users and environment interactions. Empirical tests with both artificial and real-world malware indicated that the additional power consumed by such apps was too small to be detectable with the mean error-rates of state-of-the-art measurement tools.

A similar approach was tested on an embedded medical device and a compounder by S. Clark et. al. [12]. Supervised machine learning techniques (i.e., 3-nearest neighbor, multilayer perceptron, and random forest) were used to model permissible behavior and detect deviations. Experimental results showed that the three techniques provide high detection rates, between 93.6% and 94.4%, for detecting different types of malware.

A power fingerprinting approach was presented by C. González et al. [13]. This approach relied on extracting distinctive power consumption signatures and then used pattern recognition techniques to determine if they matched expected behaviors. Preliminary results showed the feasibility of this approach. This research was expanded, and this method was used by the PFP firm[1], which provides a commercial product that detects anomalies on a device by analyzing its power consumption.

Our work is different from C. González et al. [13]. The primary difference is that we detect cyber-attacks by collecting and analyzing the internal DC power utilization of the compromised devices. Moreover, we monitor all the rails supplying power to the PLC, whereas they only monitored power utilization of the central processor. In our previous work [14], we examined whether malware (particularly rootkits) generates a detectable signal in the power consumption of a general-purpose computer. Unsupervised machine learning techniques were used to analyze CPU and motherboard power data measured from the power supply for detecting the Alureon rootkit with a high detection rate.

In this paper, we monitor internal DC power of a PLC for identifying different power profiles when the SCADA systems were operated normally and maliciously. Our work was based on the hypothesis that cyber-attacks would require more of the device's resources which would be reflected in power usage. To the best of our knowledge, this is the first work to collect DC power of SCADA devices for identifying cyber-attacks.

3. Experimental Design

In this section, we first introduce the SCADA testbed used for launching experiments and simulating cyber-attacks. After that, we demonstrate the approach to monitor PLC power consumption with detailed steps for hardware configuration. We also discuss the DF-1 protocol, a communication protocol for exchanging messages between the HMI and PLC, and present the data frame structure of DF-1 messages. We discovered several vulnerabilities in the DF-1 protocol, exploited these vulnerabilities, and successfully compromised the SCADA testbed system by command injection attacks, replay attacks and denial of service (DoS) attacks.

3.1 SCADA Testbed Hardware Configuration

The SCADA testbed consisted of two PCs and an Allen Bradley SLC 5/03 7 slot PLC which was hosted in a Allen Bradley CM-184 PLC Trainer. The Trainer,

[1]http://www.pfpcybersecurity.com/

designed for educational purposes, provided a panel of switches and LED lights wired with relays for interacting with the PLC. One PC was used to establish a communication with the PLC via HMI software. The HMI software was used to program, monitor, and control the PLC. The second PC was used to save the data collected by a Data Acquisition System (DAQ). Only four slots of the PLC were used in our experiments. A description of each slot is shown in Table 1. Operational commands and responses were communicated between the HMI application and the PLC through an RS232 cable. The main components of the SCADA testbed are shown in Figure 3.

Table 1: PLC 7-Slot description

Slot #	Description
1	8K memory SLC-5/03 controller
2	Input interface (8 switches)
3	Output interface (8 LED lights)
4	Input and output interface (1 switch and 1 LED light)
5-7	Unused

To monitor the power consumption of the PLC, we probed the power supply unit (PSU) on the PLC printed circuit board (PCB). We found four power rails in total. Two +5V rails and one +24V rail are from the backplane of the PLC. The other +24V rail is an external power supply for the output switches. Note that only one of the +5V rails and one +24V rail were required to supply the analog I/O modules of the PLC. Based on the power consumption data we monitored for the two +5V rails, we determined the +5V rail used by the CPU board. We plot the figures of power consumption data for this +5V rail and the +24V rail power supplied by the backplane of the PLC in Section 5.

We used a 16-bit multifunction data acquisition (DAQ) device [15] to collect digital power consumption data of these four rails in real-time. The DAQ provides power data with timestamps, is able to sample at a rate of 250KHz, and can monitor up to 16 channels. The DAQ was attached to the PLC's power supply through a DC current sensor board (described later) which provided both voltage and current data. For these experiments, we only monitored the three PCB voltage rails (both of the +5V rails and one of the +24V rails). The other +24V rail was on a different ground and confused the results when we included it.

The main challenge we encountered while monitoring the current consumption of the PLC was that the sensors we used in [14] were not sensitive enough to obtain clear current signals in the mA range. To address this challenge, a different breakout board (INA169 analog DC current sensor) was used. The INA169 sensor [16] is a "high-side current monitor", which means that a shunt resistor is placed on the positive power rail and measures the voltage drop across that resistor. This sensor thus provides both voltage and current data. Figure 1 shows the analog DC current sensor that was used to obtain the current consumption, while Figure 2 shows the hardware configuration that was used for our experiments.

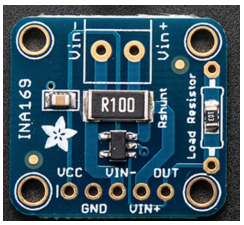

Figure 1: INA169 analog DC current sensor

Figure 2: Hardware configuration

3.2 DF-1 Protocol

The DF-1 protocol is an asynchronous byte-oriented protocol used to communicate between the HMI and the PLC via the RS232 link [17]. Figure 3 shows our complete SCADA testbed. Legitimate users can control the SCADA testbed by sending commands to the PLC through the HMI. RSLogix 500 [18] was the HMI application we used for interacting with the PLC.

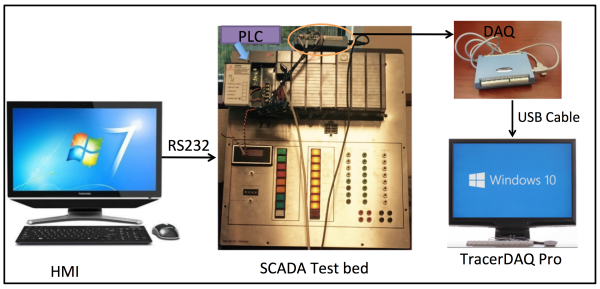

Figure 3: SCADA testbed

The data in the messages sent between the HMI and the PLC were DF-1 protocol hex data in full duplex. As a consequence, commands sent from the HMI to the PLC were transmitted as hex bytes in the data-link layer. To simulate cyber-attacks on the SCADA testbed, reverse engineering was performed on the hex byte frames. Figure 4 shows a single data-link layer command message frame of the DF-1 full duplex protocol that was transmitted from the HMI to the PLC, Table 2 shows a description for each field of this message. See [17] for more details on the DF-1 protocol.

Our test PLC had a physical key on the front for switching between Program, Run, and Remote modes. We assumed this would provide a hard interlock preventing switching modes.

The DF-1 protocol contains commands for switching modes. An example of a malicious code that could be used by an attacker to change the PLC Run mode to Program mode is 0x 10 02 01 00 0f 00 01 00 80 **01** 10 03 8f 29. An explanation of the message frame can

also be found in Table 2. Note that the highlighted 0x01 character after "FNC" (0x80) is the optional Mode field. Here 0x01 means changing the current mode to the Program mode; 0x06 would place the PLC into Run mode.

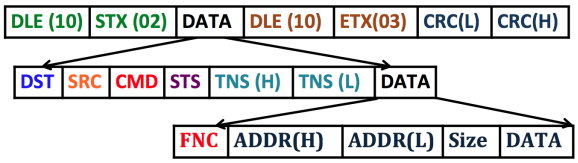

Figure 4: DF1 Full duplex data-link layer message frame

Table 2: Meaning of DF-1 full duplex data-link layer message frame

Field/Symbol	Meaning/Content	Example Packet
DLE STX	Sender symbol separating the multi multi-drop header from data	0x10 0x02
DST	Destination node for the message	0x01
SRC	Source node for the message	0x00
CMD	Command Code	0x0f
STS	Status Code	0x00
TNS	transaction number (2 bytes)	0x01 0x00
FNC	Function Code	0x80
ADDR	Address of memory location (2 bytes)	0x10 0x03
Size	Number of bytes to be transferred	NA
Data	Data values transferred by the message	NA
DLE ETX CRC	Sender symbol terminating a message	0x10 0x03 0x8f 0x29
CRC	check characters (2 bytes)	0x8f 0x29

3.3 Vulnerabilities of SCADA Systems

The forms of attacks against SCADA systems are different than those for general PCs. Vulnerabilities residing in the HMI computer, communication channels between the HMI and PLC, and software can be exploited to compromise SCADA systems. Forms of SCADA attacks include:

- *Communication Vulnerabilities*: Messages sent by the DF-1 protocol are not encrypted or protected. Hex byte messages can be easily monitored and fabricated. Examples of types of communication vulnerabilities are:
  - *Sniffing*: A Serial Protocol Analyzer tool can be used to monitor and capture unencrypted communications transmitted between the HMI and the PLC.
  - *Command Injection*: Attackers send malicious commands to change PLC operations. As an example, changing the PLC Run mode to Program Mode can be easily achieved by sending the message shown in Figure 4.
  - *Response Injection*: Attackers modify the reply messages sent from the PLC to the HMI. By monitoring normal response messages, attackers could modify the bytes in order to deceive the HMI as to the PLC's true status and vice versa.
  - *Replay*: Attackers send repeated commands to the PLC in order to control the system. Attackers first sniff legitimate commands that were sent from the HMI to the PLC, then record these commands and resend them to the PLC in a malicious way. This can be to confuse or disable the device.
- Spoofing: Introducing software on the controlling PC that mimics the HMI software.
- Denial of Service: An attacker sends various commands via the RS232 link exhausting the PLC's CPU resources.

For our testbed system, as a Response Injection-type attack, we composed a Python script that masqueraded as the RSLogix 500 to the PLC with the objective to interject itself in between. As a consequence, the response messages from the PLC were captured by the Python script rather than being sent directly to the HMI. This demonstrated a successful Man-in-the-Middle (MITM) insertion that would allow the HMI and PLC to be manipulated independently.

4. Data Collection

To collect precise data, a measurement computing data acquisition system (DAQ) was used. The DAQ device collected analog power consumption data with the help of INA169 sensors, and converted analog data to digital data which was sent to the data collection PC running the *TraceDAQ Pro* application via a USB cable. TracerDAQ Pro is an out-of-the box virtual instrument that displays and analyzes data and general signals with a customized sampling time [19]. This tool ran on a separate PC from the one running the HMI software in order to provide reliability during the experimentation process. Data was saved as a CSV file. The sampling time for collecting the power consumption data for these experiments was one second.

We first collect power consumption data under two normal scenarios. The first normal scenario had one closed switch lighting one LED light. The second normal scenario had one closed switch lighting all eight LED lights. This was a sanity check that demonstrated that our power measurement system functioned correctly and could measure the difference in power between these two normal scenarios.

The three attack scenarios we tested are command injection attacks, replay attacks and denial of service attacks, which were introduced in Section 3.3. For command injection we remotely changed the operation of the PLC to switch from run mode to program

mode. This would allow the PLC to be unknowingly reprogrammed by an attacker with enough knowledge of the PLC. For the denial of service attack we sent a large amount of communication packets to exhaust the PLC's resources. For the replay attack, sequences of legitimate commands were captured, modified and repeated to the PLC. Python scripts were written to simulate these cyber-attacks on the experimental machine.

5. Experimental Results

As part of the data pre-processing, the voltage and current for each of the monitored rails were multiplied to obtain the power consumption of the PLC. The next step was to compare normal behavior with the abnormal behavior.

Our primary goal was to prove that there is a detectable difference in the power profile between normal behavior, termed *normal*, and behavior when under attacks, termed *abnormal*. To do this, we plotted the normal data against the abnormal behavior.

5.1 Analysis for the +5V and +24V Rails

Several comparisons were made as part of our data analysis. The first scenario tested was to verify the power consumption of the PLC when one light was turned on versus when all eight lights of the PLC Trainer were on.

From the rails that were monitored, the +24V rail and one of the +5V rails were the ones that showed an increase in the power usage when all the lights of the PLC Trainer were cycled. Figure 5 shows a comparison of the PLC's power usage for one light versus when all the lights of the PLC were on for the +24V rail, while Figure 6 shows the same comparison for one of the +5V rails.

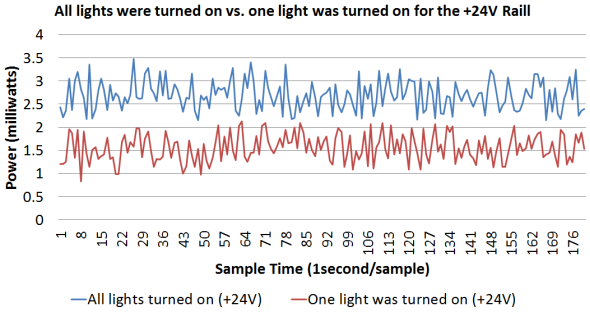

Figure 5: Comparing PLC'S power usage when all lights were turned on vs. when one light was turned on for the +24V rail

As expected, the power usage of the PLC increases when all eight lights were turned on in comparison to when only one light was turned on. Note that the +24V rail monitored is being used by the PLC and is not supplying voltage to the lights themselves. We designed this scenario as a confirmation of our system to measure

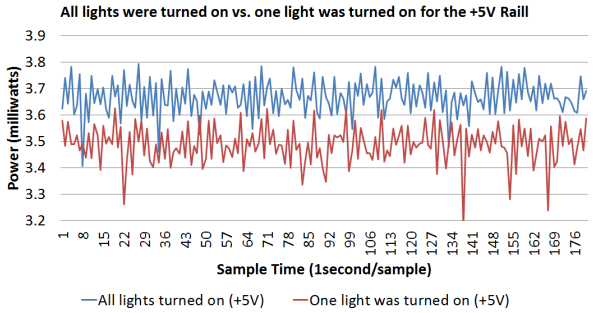

Figure 6: Comparing PLC'S power usage when all lights were turned on vs. when one light was turned on for one of the +5V rails

the difference when a larger amount of power (switching more lights) was obviously occurring. Even in this nascent phase, this type of comparison could be used as a second-party verification system for a monitored SCADA system to assure it is switching as it was designed.

For the cyber-attack scenarios, we compared normal power usage of the PLC (one light of the PLC was turned on) with the power usage when cyber-attacks were simulated. The command injection attack changes the mode of the PLC from running mode to program mode. In other words, emulating malicious users attempting to disable a PLC, our Python scripts would, over a ten second cycle, change the mode from run mode to program mode after two seconds, then pause for eight seconds before switching back. Therefore, the PLC was in the program mode and was waiting for new commands. We did not send new commands, but kept the PLC waiting for eight seconds and then changed the PLC mode back to the running mode, in which mode the PLC performed normally again. This was repeated and power data was collected. Figure 7 and Figure 8 show a comparison of the PLC's power usage when one light was turned on versus the command injection attack for the +24V and +5V rails.

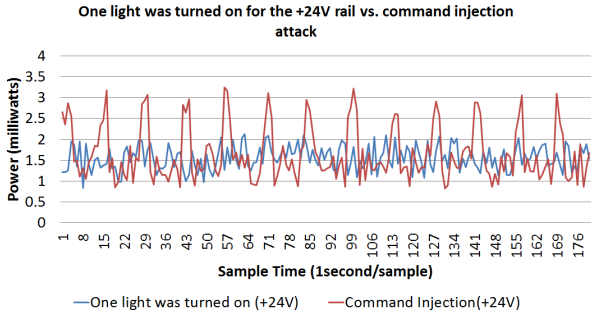

Figure 7: Comparing PLC'S power usage when one light was turned on vs. command injection attack for the +24V rail

As we can see from the figures, when the PLC operated normally, in which case all eight lights were flash-

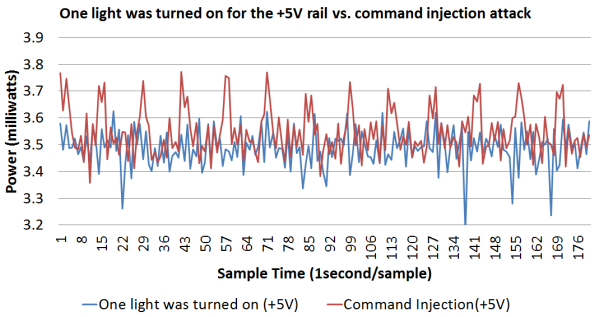

Figure 8: Comparing PLC'S power usage when one light was turned on vs. command injection attack for the +5V rail

ing, the power consumption of the +24V rail was between 3 to 3.5 milliwatts. When the PLC mode was changed to the program mode, the power consumption dropped to 1 milliwatts, which was even lower than the power consumption of the normal scenario when one light was turned on. Similar patterns can be found in Figure 8. This experiment validated that the power consumption feature can be adopted to easily distinguish normal scenarios and command-injection attacks.

We also simulated a Denial of Service (DoS) attack and collected power consumption data of the PLC. In this scenario, malicious users kept sending a large amount of diagnostic node requests to read a block of status information from the RAM of PLC. The PLC was therefore busy with receiving messages and replying with status information. Since the PLC processor only allows up to 4,096 inputs and outputs and the instruction memory capacity is only 8K, the PLC resources were exhausted immediately after DoS attacks were launched, thus DoS attacks prevented legitimate usage of the PLC. Figure 9 and Figure 10 show a comparison of the PLC's power usage when one light was turned on versus +24 and +5 rails power usages when the PLC was under the DoS attack.

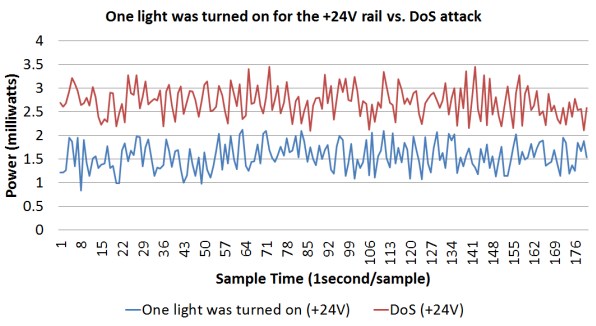

Figure 9: Comparing PLC'S power usage when one light was turned on vs. DoS for the +24V rail

From the figures we can see that the PLC power consumption was much higher than the normal scenario. Although we can distinguish the power profiles of DoS

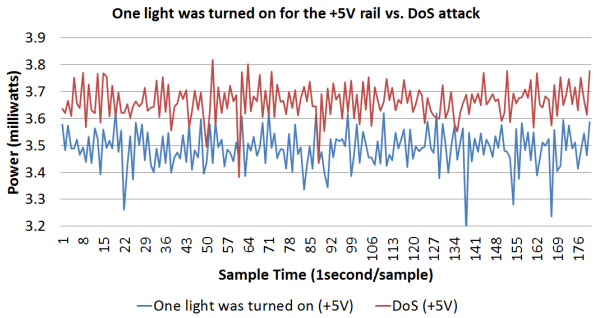

Figure 10: Comparing PLC'S power usage when one light was turned on vs. DoS for the +5V rail

attacks and one light on, when comparing the DoS data with another normal case when all lights turned on, the differences are not pronounced.

Finally, we emulated the replay attack. In this scenario, malicious users sniffing normal commands sent from the HMI to the PLC and resent them to the PLC to retrieve useful information. As shown in Figure 11 and Figure 12, replay attack power consumption data is much higher than the normal case with one light turned on.

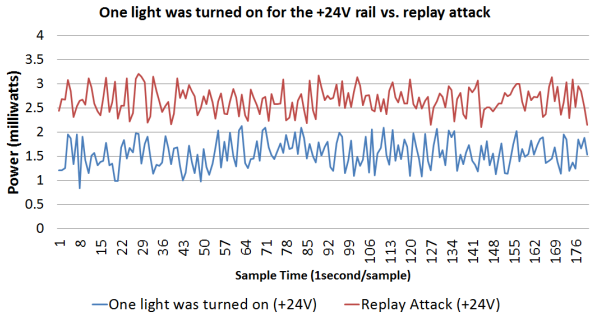

Figure 11: Comparing PLC'S power usage when one light was turned on vs. replay attack for the +24V rail

Therefore, we can easily distinguish replay attacks from the normal scenario. Note that we did not compare various attack benchmarks but comparing each attack benchmark with the normal benchmark since this paper aims to prove that there is a detectable difference between SCADA normal and abnormal operations. In the future, we will compare attack data and use unsupervised techniques to identify and classify attacks regarding to the PLC power consumption.

6. Conclusion

This paper presents how we developed a SCADA testbed with the objective to detect cyber-attacks by monitoring and analyzing the power consumption of a PLC. The power consumption of the PLC was monitored under three attack scenarios: command injec-

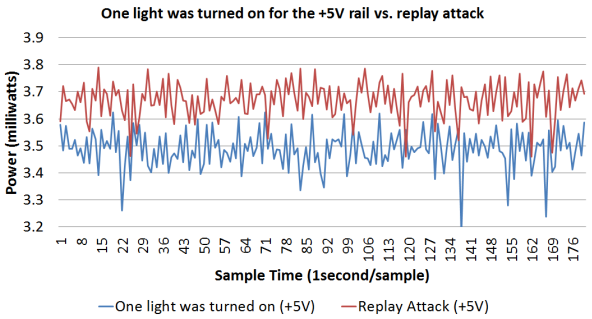

Figure 12: Comparing PLC'S power usage when one light was turned on vs. replay attack for the +5V rail

tion, DoS, and replay. Results shown that these cyber-attacks leave a detectable signal on the power consumption of a PLC.

For the cases that were difficult to distinguish, we plan to apply new techniques, which we have successfully applied on general-purpose computers to detect minute changes in power signals caused by malware. We are also planning to apply unsupervised machine learning techniques to discern these signals. Machine learning can take advantage of the the repetitive nature of PLCs (i.e., they generally perform a few tasks repeatedly) in normal operation. Furthermore,we will also attempt to characterize the types of cyber-attacks using machine learning techniques.

7. References

[1] "Report on Cybersecurity and Critical Infrastructure in the Americas." Retrieved in September 2016 from http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/reports/critical-infrastructures-west-hemisphere.pdf.

[2] "Dell Security Annual Threat Report 2015." Retrieved in September 2016 from https://software.dell.com/docs/2015-dell-security-annual-threat-report-white-paper-15657.pdf.

[3] B. Miller and D. Rowe, "A survey SCADA of and Critical Infrastructure Incidents," in *Proceedings of the 1st Annual Conference on Research in Information Technology*, RIIT '12, (New York, NY, USA), pp. 51–56, ACM, 2012.

[4] J. Norman, "The First Malware to Spy on and Subvert Industrial Systems," 2010.

[5] D. H. J. M. Aleksandr Matrosov, Eugene Rodionov, "Stuxnet Under the Microscope," *Epistemics in Science, Engineering and Technology (ESET)*, 2011.

[6] H. Stark, "Mossad's Miracle Weapon: Stuxnet Virus Opens New Era of Cyber War," 2011. Retrieved in August 2016 from http://www.spiegel.de/international/world/mossad-s-miracle-weapon-stuxnet-virus-opens-new-era-of-cyber-war-a-778912.html.

[7] R. V. Nell Nelson, "The Impact of Dragonfly Malware on Industrial Control Systems," *SANS Institute*, 2016.

[8] Q. Chen and S. Abdelwahed, "A Model-Based Approach to Self-Protection in SCADA Systems," in *9th International Workshop on Feedback Computing*, USENIX Association, June 2014.

[9] Q. Chen, S. Abdelwahed, and A. Erradi, "A Model-Based Validated Autonomic Approach to Self-Protect Computing Systems," *Internet of Things Journal, IEEE*, vol. 1, pp. 446–460, Oct 2014.

[10] H. Bao, R. Lu, B. Li, and R. Deng, "BLITHE: Behavior Rule-Based Insider Threat Detection for Smart Grid," *IEEE Internet of Things Journal*, vol. 3, no. 2, pp. 190–205, 2016.

[11] J. Hoffmann, S. Neumann, and T. Holz, "Mobile Malware Detection Based on Energy Fingerprints: A Dead End?," in *Research in Attacks, Intrusions, and Defenses*, Springer, 2013.

[12] S. S. Clark, B. Ransford, A. Rahmati, S. Guineau, J. Sorber, W. Xu, K. Fu, A. Rahmati, M. Salajegheh, D. Holcomb, *et al.*, "Wattsupdoc: Power Side Channels to Nonintrusively Discover Untargeted Malware on Embedded Medical Devices," in *HealthTech*, 2013.

[13] C. R. A. Gonzalez and J. H. Reed, "Power Fingerprinting in SDR and CR Integrity Assessment," in *Military Communications Conference, 2009. MILCOM 2009. IEEE*, pp. 1–7, 2009.

[14] J. M. Hernández, R. Bridges, J. Nichols, S. Prowell, and K. Goseva-Popstojanova, "Towards a Malware Detection Framework Based on Power Consumption Monitoring," 2016. Retrieved in September 2016 http://www.ieee-security.org/TC/SP2016/poster-abstracts/46-poster_abstract.pdf.

[15] "USB-1608g Series 16-bit High-Speed Multifunction DAQ Devices." Retrieved in August 2016 from http://www.mccdaq.com/usb-data-acquisition/USB-1608G-Series.aspx.

[16] "INA169 Analog DC Current Sensor Breakout - 60v 5a Max." Retrieved in September 2016 from https://www.adafruit.com/product/1164.

[17] Allen-Bradley, "DF-1 Protocol and Command Set Reference Manual." Retrieved in August 2016 from http://docplayer.net/224339-Allen-bradley-df1-protocol-and-command-set-reference-manual.html.

[18] "Rockwell Software RSLogix 500." Retrieved in September 2016 from http://www.rockwellautomation.com/rockwellsoftware/products/rslogix500.page.

[19] "TracerDAQ Pro." Retrieved in August 2016 from http://www.mccdaq.com/daq-software/tracerdaq-pro.aspx.